이동 로봇의 자율 주행용 함수 개발 및 구현

정석기* · 고낙용** · 김태균***

Development and Implementation of Functions for Mobile Robot Navigation

Seok-Ki Jeong* · Nak-Yong Ko** · Tae-Gyun Kim***

요 약

본 논문은 이동 로봇의 자율 주행을 위한 중요한 기술 중 하나인 위치 추정을 위한 함수 구현에 관해 서술되었다. 기존의 로봇 자율주행용 함수 라이브러리 중 일부는 모의실험에만 사용할 수 있기 때문에 실제 적용에 제한이 따른다. 본 논문은 실내 이동 로봇의 위치 추정을 위해 사용할 수 있는 함수의 개발에 중점을 두었다. 함수들은 추측항법, 이동 로봇의 운동 모델, 거리 측정 센서의 측정 모델, 그리고 빈번히 사용되는 방향 관련 연산에 대해 구현되었다. 구현된 함수들은 다양한 로봇과 센서에 적용할 수 있다. 사용자는 적절한 함수를 선택하여 로봇 운동과 센서 측정 불확실성의 다양한 유형을 구현할 수 있다. 구현된 함수들은 모의실험과 실제 실험을 통해 시험 및 증명되었다.

ABSTRACT

This paper describes implementation of functions for mobile robot localization, which is one of the vital technologies for autonomous navigation of a mobile robot. There are several function libraries for mobile robot navigation. Some of them have limited applicability for practical use since they can be used only for simulation. Our research focuses on development of functions which can be used for localization of indoor robots. The functions implement deadreckoning and motion model of mobile robots, measurement model of range sensors, and frequently used calculations on angular directions. The functions encompass various types of robots and sensors. Also, various types of uncertainties in robot motion and sensor measurements are implemented so that the user can select proper ones for their use. The functions are tested and verified through simulation and experiments.

키워드

mobile robot, localization library, autonomous navigation, differential drive robot, bicycle robot model
이동 로봇, 위치 추정 라이브러리, 자율주행, 차륜 구동 로봇, 이륜차 로봇 모델

Ⅰ. 서 론

자율주행 기술은 로봇이 주변 환경을 파악하는 지도 작성, 로봇 스스로의 위치를 파악하는 위치 인식 기술, 목적지까지의 경로를 결정하기 위한 경로 계획 기술, 그리고 장애물을 피하기 위한 회피기술의 구현이 필요하다. 이 기술 중 위치 인식 기술은 가장 중요하다. 로봇이 자신의 위치를 알지 못한다면 목적지까지 어떤 경로로 주행해야 하는지 알 수 없다. 그러므로 위치 인식 기술은 이동 로봇에서 필수적 핵심

* 조선대학교 제어계측공학과(seokki@chosun.kr)
** 교신저자 : 조선대학교 제어계측공학과 교수(nyko@chosun.ac.kr)
*** 조선대학교 제어계측공학과(ktg9114@naver.com)
접수일자 : 2013. 02. 21 심사(수정)일자 : 2013. 03. 15 게재확정일자 : 2013. 03. 22

기술이다[1]. 위치 추정 기술에 대한 연구는 다양한 알고리즘들과 연구자의 이해에 따른 여러 방법으로 구현되었다. 위치 추정 기술 연구의 문외한 또는 입문자는 기존에 구현된 기술의 자세한 해설이 없거나 입력 및 출력 정보에 대한 설명이 없어 어려움을 겪는다. 본 연구를 통해 주행기술의 위치 추정에 관한 함수들을 보이고 이 기능을 사용하기 위한 기본이론과 입출력정보에 대해 설명한다.

이동 로봇의 자율주행을 위해 사용 가능한 라이브러리는 MRPT(Mobile Robot Programming Toolkit)[2-4], Robotics Toolbox for Matlab[5][6], uRON(Universal Robot Navigation)[2][7], CARMEN(Carnegie Mellon Robot Navigation Toolkit)[2][8], 그리고 Karto SDK[2][9] 등이 있다. 각각의 라이브러리는 적용된 알고리즘, 작업환경, 그리고 사용방법 등의 제약으로 사용자가 로봇에 적용하기에 어려움을 지닌다. 그림 1은 주행로봇에 사용가능한 라이브러리를 비교한 그림이다[2].

Short Name	uRON	CARMEN	Karto SDK	MRPT
Full Name	Universal Robot Navigation	Carnegie Mellon Robot Navigation Toolkit	Karto SDK	Mobile Robot Programming Toolkit
Homepage	http://robotsk.etri.re.kr/wiki/uRON	http://carmen.sourceforge.net/	http://www.kartorobotics.com/	http://www.mrpt.org/
Developer	ETRI, Republic of Korea	Open Source (CMU, USA)	Karto Robotics (SRI International, USA)	Open Source (Univ. of Malaga, Spain)
Licence	-	GPL	LGPL/Commercial	GPL
First Version	1.0(2009. 12. 18.)	0.1(2004. 10. 15.)	1.0.0(2008. 12. 10.)	0.5.3(2008. 1. 31)
Recent Version	1.5(2010. 8. 26.)	0.7.4-beta(2008. 10. 23.)	2.0.1(2010. 7. 22.)	0.9.2(2010. 10. 28.)
Language	C++	C	C++	C++
Supported Languages	-	C++, Java, Python	C#	-
Operating Systems	Windows/Linux	Linux	Windows/Linux/Mac	Windows/Linux/Mac
Map	<Grid Map> Binary Grid Map Occupancy Grid Map <Topology Map> Road Map	Occupancy Grid Map (maptools) Hierarchial Map(hmap)	Occupancy Grid Map (karto::OccupancyGrid) Incremental Occupancy Grid Map (karto::IncrementalOccupancyGrid)	<Metric Map> (mrpt::slam::CMetricMap) Occupancy Grid Map Multiple Metric Map Hierarchial Metric Map (HMT-SLAM Map) Graph-SLAM Map
Localization	EKF Localization	MCL	MCL (karto::localiner::Localiner)	MCL

그림 1. 주행로봇용 라이브러리
Fig. 1 Libraries for mobile robot

Robotics Toolbox for Matlab은 Matlab을 이용하여 모의실험을 할 수 있도록 만든 라이브러리이다. 이 라이브러리를 이용한 모의실험으로는 추측 항법, 확장 칼만 필터 알고리즘을 적용한 위치 추정, 랜드마크(Land Mark)의 위치를 찾는 지도 작성, 확장 칼만 필터를 적용한 SLAM(Simultaneous Localization and Mapping), 그리고 몬테카를로 알고리즘(Monte Carlo Algorithm)을 이용한 위치 추정 등이 있다. 이와 같이 Robotics Toolbox for Matlab은 위치 추정에 관한 다양한 모의실험을 할 수 있는 장점이 있다. 하지만 Robotics Toolbox for Matlab은 사용자의 목적에 부합되는 모의실험을 위해 소스 코드의 분석과 수정의 필요성을 단점으로 갖는다. MRPT는 이동 로봇의 연구에 자주 사용되는 라이브러리이다. 그러나 MRPT는 라이브러리 구조가 복잡하여 사용자가 필요한 기술을 검색하기에 어렵다. 또한 사용자가 필요한 함수를 찾더라도 그와 연결된 함수 및 입출력 정보들에 대한 파악의 필요성을 단점으로 갖는다[10].

본 논문에서는 기존의 라이브러리들이 가진 단점을 보완하기 위해 구현된 함수의 배경이론과 입출력 정보를 기술하고 있다. 본 연구를 통해 구현된 함수는 모의실험 또는 특정 로봇이 아닌 다양한 로봇에 적용 가능하다. 그리고 구현된 함수는 자율 주행의 위치 추정에 관련된 로봇 알고리즘 구현 시 알고리즘 구현의 시간소모를 줄이는 데 도움이 된다.

본 논문의 2장과 3장에서는 차륜 구동 로봇과 이륜차 로봇 모델에 적용 가능토록 구현된 추측 항법과 운동 모델 함수를 설명한다. 4장에서는 외부 환경 정보를 인지하기 위한 측정 모델 함수에 대해 설명한다. 5장에서는 방향의 연산에 대한 함수를 설명하고 6장에서는 함수를 이용한 모의실험과 실제 실험에 대해 기술한다. 마지막 7장에서는 본 논문의 결론을 맺는다.

II. 추측 항법

추측 항법은 이전 위치 정보와 로봇의 내부정보를 이용하여 필요한 시각에서 로봇의 위치를 추정하는 방법이다. 즉, 추측 항법은 로봇의 이전 위치 정보와 로봇의 내부정보를 이용하여 일정시간이 지난 후 위치를 추정한다. 식 2.1은 시각 t에서 로봇의 위치를 나타내는 위치 정보인 x_t, y_t, θ_t를 갖는 X_t에 대한 식이다. x_t, y_t는 직교좌표상의 로봇의 좌표 값이다. 그리고 θ는 직교좌표의 x축과 로봇의 방향이 이루는 사잇각을 나타낸다.

$$X_t = [x_t\, y_y\, \theta_t] \tag{2.1}$$

로봇의 내부정보에는 대표적인 예로 모터의 엔코더 데이터를 들 수 있다. 로봇의 속도정보는 모터의

엔코더 데이터를 이용하여 구할 수 있다. 획득한 속도정보와 주행시간을 이용하여 위치를 추정할 수 있다.

2.1 차륜 구동 로봇

차륜 구동 로봇은 좌우 바퀴를 제어하여 주행하는 로봇을 의미한다. 그림 2는 차륜 구동 로봇구조와 로봇의 운동에 대해 나타내고 있다. 로봇의 두 바퀴의 반지름은 r로 같고, 두 바퀴의 중심 사이 거리는 d, 좌측 바퀴의 주행속도는 v_L, 그리고 우측 바퀴의 주행속도는 v_R임을 보이고 있다. 그리고 로봇의 속도정보는 주행속도는 v, 회전속도는 w로 나타내고 있다. 그리고 *ICR(instantaneous center of rotation)*는 이동 로봇이 v와 w의 속도로 주행할 경우 이전 위치에서 로봇의 두 바퀴의 중심을 지나는 연장선과 현재 위치에서 로봇의 두 바퀴의 중심을 지나는 연장선이 교차하는 지점이다[11].

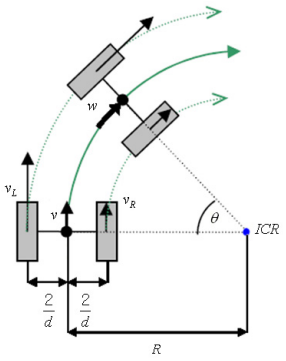

그림 2. 차륜 구동 로봇 운동 예
Fig. 2 The motion example of differential drive mobile robot

좌측 바퀴와 우측 바퀴의 이동속도는 식 2.2와 같다. 좌우 바퀴의 이동속도는 호의 길이가 반지름과 라디안 단위인 중심각의 곱과 같음을 이용한다. 로봇 좌우 바퀴의 각속도 w_L과 w_R은 모터의 엔코더 데이터를 이용하여 구할 수 있다. w는 *ICR*을 중심으로 로봇의 회전변화를 나타내는 로봇의 회전속도이다. 식 2.3은 식 2.2와 같은 방법으로 유도된다. 식 2.4의 이동속도 v와 회전속도 w는 식 2.3에서 v_L과 v_R에 대해 나타낸 두 식을 이용하여 유도된다[11].

$$v_L = rw_L \qquad (2.2)$$
$$v_R = rw_R$$

$$wR = v \qquad (2.3)$$
$$w(R+\frac{d}{2}) = v_L$$
$$w(R-\frac{d}{2}) = v_R$$

$$v = \frac{v_L + v_R}{2} \qquad (2.4)$$
$$w = \frac{v_R - v_L}{d}$$

로봇 좌우 바퀴의 각속도 w_L, w_R과 두 모터의 중심 사이의 거리 d를 획득 가능한 경우 식 2.2부터 2.4을 통해 로봇의 직진속도 v와 회전속도 w를 구할 수 있다[12].

표 1은 차륜 구동 로봇의 추측 항법인 *DR_DD*의 의사코드를 나타낸다. 구현된 함수는 입력정보로 이전 위치 정보 X_{t-1}, 직진속도 v와 회전속도 w, 그리고 시간 간격 Δt를 입력받는다. 함수의 결과는 표의 연산 결과인 추정된 위치 정보 X_t를 반환한다.

표 1. 차륜 구동 로봇의 추측 항법 함수
Table 1. Dead reckoning function of differential drive robot

Algorithm $DR\_DD(X_{t-1},\ v,\ w,\ \Delta t)$
1: *if* $w \neq 0$
2: $\quad x_t = x_{t-1} - \frac{v}{w}(\sin\theta_{t-1} - \sin(\theta_{t-1} + w\Delta t))$
3: $\quad y_t = y_{t-1} + \frac{v}{w}(\cos\theta_{t-1} - \cos(\theta_{t-1} + w\Delta t))$
4: $\quad \theta_t = \theta_{t-1} + w\Delta t$
5: *otherwise*
6: $\quad x_t = x_{t-1} + v\cos\theta_{t-1}\Delta t$
7: $\quad y_t = y_{t-1} + v\sin\theta_{t-1}\Delta t$
8: $\quad \theta_t = \theta_{t-1}$
9: *return* X_t

2.2 앞바퀴 속도 정보를 이용한 이륜차 로봇

이륜차 로봇은 자전거의 구조로 설명 가능하다. 자전거의 뒷바퀴는 몸체와 고정되어 항상 같은 방향을

유지해야 하고, 자전거의 앞바퀴를 조정하여 진로 방향을 결정할 수 있어야 한다. 이륜차 로봇은 그림 3과 같이 나타낼 수 있다. 그림 3에서는 뒷바퀴의 좌표 정보인 x, y와 x축과 로봇 몸체가 이루는 각인 로봇의 지향 각 θ, 로봇몸체와 앞바퀴의 사잇각 ϕ에 대해 나타내고 있다[13].

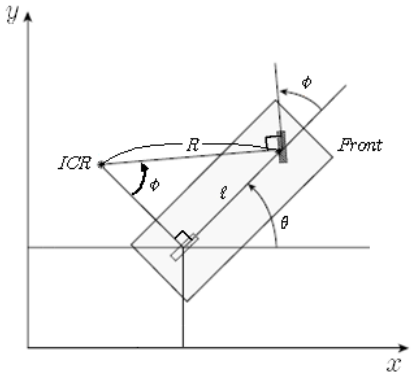

그림 3. 앞바퀴 속도 이용 이륜차 로봇
Fig. 3 Bicycle model of using driving velocity of front wheel

표 2는 앞바퀴 속도를 이용한 이륜차 로봇의 추측항법을 의사코드로 나타낸다. 함수 명칭은 *DR_BF*라 정의했다. 입력될 정보는 이전 위치 정보 X_{t-1}와 앞바퀴의 속도정보 v_f, 로봇의 몸체와 앞바퀴의 사잇각 ϕ, 로봇의 앞바퀴와 뒷바퀴 사이거리 ℓ, 그리고 시간간격 Δt가 있다. θ의 변화인 $\dot{\theta}$는 로봇 몸체의 방향과 x축과의 사잇각의 변화를 나타낸다. 이 변화는 차륜 구동 로봇의 회전속도인 w와 같으므로 *DR_DD* 함수를 재사용할 수 있다. 함수의 결과는 재사용한 *DR_DD* 함수에서 획득한 위치 정보 X_t를 반환한다.

표 2. 앞바퀴 속도 이용 이륜차 로봇 추측 항법 함수
Table 2. Dead reckoning function of bicycle model using front wheel driving velocity

Algorithm DR_BF(X_{t-1}, v_f, ϕ, ℓ, Δt)
1: $w = v_f \frac{\sin\phi}{\ell}$
2: X_t = *call DR_DD*(X_{t-1}, v_f, w, Δt)
3: *return* X_t

2.3 뒷바퀴 속도 정보를 이용한 이륜차 로봇

이륜차 로봇 모델에서는 앞바퀴와 뒷바퀴가 직교좌표계의 x축과 y축 방향으로 미끄러짐이 없다는 가정을 한다. 이는 식 2.5로 표현된다. 여기에서 직교좌표상의 앞바퀴와 뒷바퀴의 좌표를 (x_f,y_f) 그리고 (x, y)로 나타낸다. 그리고 식 2.5에서 이 좌표 값들의 변화에 대해 각각 $\dot{x}_f$, $\dot{y}_f$, $\dot{x}$, $\dot{y}$로 나타낸다. 식 2.6은 앞바퀴의 좌표 값 x_f와 y_f를 뒷바퀴의 좌표 값인 x와 y를 이용하여 나타낸다. 식 2.7은 식 2.5에 식 2.6을 적용하고 $\dot{x}$와 $\dot{y}$에 대해 정리한 식이다. 식 2.8은 식2.5의 $\dot{x}$와 $\dot{y}$에 대한 식의 비홀로노믹 구속(*nonholonomic constraint*)을 충족시킨다[13]. 비홀로노믹 시스템은 적분 불가능한 구속조건으로부터 도출되는 시스템을 의미한다[14]. 식 2.9는 식 2.7에 식 2.8을 적용하여 $\dot{\theta}$에 대해 전개한 식이다. 식 2.9를 통해 x축과 로봇 방향의 사잇각의 변화를 구할 수 있다. v_r은 뒷바퀴의 속도를 의미한다[13].

$$\begin{aligned}\dot{x}_f\sin(\theta+\phi)-\dot{y}_f\cos(\theta+\phi)&=0\\ \dot{x}\sin\theta-\dot{y}\cos\theta&=0\end{aligned} \tag{2.5}$$

$$\begin{aligned}x_f&=x+\ell\cos\theta\\ y_f&=y+\ell\sin\theta\end{aligned} \tag{2.6}$$

$$\dot{x}\sin(\theta+\phi)-\dot{y}\cos(\theta+\phi)-\dot{\theta}\ell\cos\phi=0 \tag{2.7}$$

$$\begin{aligned}\dot{x}&=v_r\cos\theta\\ \dot{y}&=v_r\sin\theta\end{aligned} \tag{2.8}$$

$$\dot{\theta}=v_r\frac{\tan\phi}{\ell} \tag{2.9}$$

표 3은 뒷바퀴 속도를 이용한 이륜차 로봇의 추측항법을 의사코드로 나타낸다. 함수 명칭은 *DR_BR*라 정의한다. 구현된 함수는 이전 위치 정보 X_{t-1}, 뒷바퀴의 속도정보 v_r, 로봇의 몸체와 앞바퀴의 사잇각 ϕ, 로봇의 앞바퀴와 뒷바퀴 사이거리 ℓ, 그리고 시간간격 Δt를 입력받는다. 식 2.8에서 θ의 변화인 $\dot{\theta}$는 로봇 몸체의 방향과 x축과의 사잇각의 변화를 나타낸다. 이는 차륜 구동 로봇의 회전속도인 w와 같으므로 *DR_DD* 함수를 재사용할 수 있다. 반환 값은

DR_DD 함수를 통해 획득한 추정된 위치 정보 X_t를 출력한다.

표 3. 뒷바퀴 속도 이용 이륜차 로봇 추측 항법 함수
Table 3. Dead reckoning function of bicycle model using rear wheel driving velocity

Algorithm DR_BR(X_{t-1}, v_r, ϕ, ℓ, Δt)

1: $w = v_r \frac{\tan\phi}{\ell}$

2: X_t = *call DR_DD*(X_{t-1}, v_r, w, Δt)

3: *return* X_t

III. 운동 모델

추측 항법에서는 로봇의 내부정보만으로 로봇의 위치를 예측했다. 추측 항법은 로봇의 이상적인 이동에 대한 위치 추정이다. 추측 항법으로 추정된 위치 정보는 실제 환경의 로봇 위치와 일치할 수 없다. 로봇이 이동 시 갖는 오차로 인한 실제 이동 속도가 다르기 때문이다. 따라서 로봇의 이동 속도는 로봇의 속도에 대한 실험으로 획득된 오차의 적용이 필요하다. 실제 환경에서 주행하는 로봇은 많은 오차들을 갖는다. 로봇이 갖게 되는 오차는 모터의 엔코더로부터 얻게 되는 속도정보의 오차, 바퀴와 바닥면의 마찰력에 의한 미끄러짐에 대한 오차, 그리고 바퀴의 공기압으로 인한 바퀴 반지름의 오차 등이 있다. 추측 항법은 오차의 지속적인 누적을 문제점으로 갖는다. 이 문제점을 보완하기 위해 운동 모델은 로봇의 내부정보로부터 획득한 속도정보에 오차를 적용하고 적용된 속도정보를 이용해 위치를 예측하는 방법이다[15].

3.1 정규분포

운동 모델에 적용되는 속도정보의 오차는 정규분포를 이용하여 획득한다. 표 4는 정규 확률 분포를 따르는 확률 변수를 구하는 의사코드를 나타낸다. 표 4의 b는 실제 실험을 통해 얻은 운동에러의 표준편차를 적용한다. 표 4의 *rand* 함수는 b에서 $-b$까지 균등 분포를 따르는 임의의 값을 출력하는 함수이다[14]. 함수의 명칭은 *NormalDistribution*이라 정의한다.

표 4. Normal distribution 함수
Table 4. Normal distribution function

Algorithm NormalDistribution(b)

return $\frac{1}{2}\sum_{i=1}^{12} rand(b, -b)$

3.2 차륜 구동 로봇

차륜 구동 로봇은 2.1절에서 소개된 것으로 좌우 바퀴를 제어하여 주행하는 로봇을 의미한다. 차륜 구동 로봇에 대해 그림 2에서 나타내고 있다. 차륜 구동 로봇의 직진 속도에 영향을 미치는 오차정보에는 α_1, α_2가 있다. 그리고 회전 속도에 영향을 미치는 오차정보에는 α_3, α_4가 있고 로봇의 방향각에 대한 오차 정보에는 α_5, α_6이 있다. 식 3.1은 오차정보들을 갖는 A를 나타낸다. A는 로봇의 직진속도와 회전속도, 그리고 로봇의 방향에 대한 오차를 갖는다. 표 5의 차륜 구동 로봇의 운동 모델 함수는 *MM_DD*라 정의하고 의사코드로 표현한다[15]. 함수의 입력정보는 이전 위치 정보 X_{t-1}과 속도정보 v와 w, 시간 간격 Δt, 그리고 오차정보인 A를 입력받는다. *MM_DD* 함수의 출력은 위치 정보 X_t를 반환한다. 표 5의 *sample* 함수는 3.1절의 *NormalDistribution* 함수를 의미한다.

$$A = [\alpha_1\ \alpha_2\ \alpha_3\ \alpha_4\ \alpha_5\ \alpha_6] \tag{3.1}$$

표 5. 차륜 구동 로봇의 운동 모델 함수
Table 5. Motion model function of differential drive robot

Algorithm MM_DD(X_{t-1}, v, w, Δt, A)

1: $\hat{v} = v + sample(\alpha_1|v| + \alpha_2|w|)$

2: $\hat{w} = v + sample(\alpha_3|v| + \alpha_4|w|)$

3: $\hat{\gamma} = sample(\alpha_5|v| + \alpha_6|w|)$

4: $x_t = x_{t-1} - \frac{\hat{v}}{\hat{w}}\left(\sin\theta_{t-1} - \sin\left(\theta_{t-1} + \hat{w}\Delta t\right)\right)$

5: $y_t = y_{t-1} + \frac{\hat{v}}{\hat{w}}\left(\cos\theta_{t-1} - \cos\left(\theta_{t-1} + \hat{w}\Delta t\right)\right)$

6: $\theta_t = \theta_{t-1} + \hat{w}\Delta t + \hat{\gamma}\Delta t$

9: *return* X_t

3.3 앞바퀴 정보를 이용한 이륜차 로봇

표 6은 앞바퀴 속도를 이용한 이륜차 로봇의 함수를 의사코드로 나타낸다. 함수의 명칭은 *MM_BF*라 정의한다. *MM_BF* 함수의 입력정보는 이전 위치 정보 X_{t-1}와 앞바퀴의 속도정보 v_f, 몸체와 앞바퀴의 사잇각 ϕ, 로봇의 앞뒤 바퀴의 사이거리 ℓ, 시간 간격 Δt, 그리고 운동 에러 A를 입력받는다. 2.2에서 보임과 같이 로봇의 지향 각 변화인 $\dot{\theta}$는 차륜 구동 로봇의 회전속도인 w와 같다. 그러므로 *MM_DD* 함수는 *MM_DF* 함수에서 재사용 가능하다. *MM_DF* 함수의 출력 값은 위치 정보 X_t를 반환한다.

표 6. 앞바퀴 속도 이용 이륜차 로봇 운동 모델 함수
Table 6. Motion model function of bicycle model using front wheel driving velocity

Algorithm MM_BF(X_{t-1}, v_f, ϕ, ℓ, Δt, A)

1: $w = v_f \frac{\sin\phi}{\ell}$

2: X_t = *call MM_DD*(X_{t-1}, v_f, w, Δt, A)

3: *return* X_t

3.4 뒷바퀴 속도 정보를 이용한 이륜차 로봇

표 7은 뒷바퀴 속도를 이용한 이륜차 로봇의 운동 모델 함수를 의사코드로 나타낸다. 함수의 명칭은 *MM_BR*로 정의한다. *MM_BR* 함수의 입력정보는 이전 위치 정보 X_{t-1}과 뒷바퀴의 속도정보 v_r, 몸체와 앞바퀴의 사잇각 ϕ, 로봇의 앞뒤 바퀴의 사이거리 ℓ, 시간 간격 Δt, 그리고 운동 에러인 A를 입력받는다. 2.3절과 같이 로봇의 지향 각 θ의 변화인 $\dot{\theta}$는 차륜 이동 로봇의 회전속도인 w와 같다. 그러므로 *MM_BR* 함수 내에서 *MM_DD* 함수의 재사용이 가능하다. *MM_BR* 함수의 출력은 *MM_DD* 함수의 출력인 위치 정보 X_t를 반환한다.

표 7. 뒷바퀴 속도 이용 이륜차 로봇 운동 모델 함수
Table 7. Motion model function of bicycle model using rear wheel driving velocity

Algorithm MM_BR(X_{t-1}, v_r, ϕ, ℓ, Δt, A)

1: $w = v_r \frac{\tan\phi}{\ell}$

2: X_t = *call MM_DD*(X_{t-1}, v_f, w, Δt, A)

3: *return* X_t

Ⅳ. 측정 모델

측정 모델용 함수는 센서 정보와 계산된 정보를 확률분포에 적용하여 획득한 확률 값으로 신뢰도를 구한다. 본 장에서는 연구를 통해 개발된 측정 모델 함수들에 적용된 확률분포와 함수들에 대해 기술했다. 표 8은 공통으로 사용되는 확률분포를 적용한 공통 측정 모델 함수이다. 공통 측정 모델 함수의 명칭은 *MM_Common*라 정의한다. 공통 측정 모델 함수는 빛이나 초음파를 이용한 거리측정 센서의 측정 모델에 공통으로 적용되는 확률분포들로 구성된다. 그림 4는 공통으로 적용되는 확률분포들을 나타낸다[15]. *MM_Common* 함수는 입력정보로 센서에서 획득한 센서 정보 z_t^k, 계산된 정보 z_t^{k*}, 적용도 Z, 가우시안 확률분포에 사용될 표준편차σ_{hit}, 센서의 측정 가능 최댓값 $z_{\max}$을 입력받는다. 적용도는 조합할 확률분포들의 적용되는 정도를 나타낸다. 식 3.2는 적용도 Z를 나타낸다. 적용도 Z는 적용도z_{whit}, z_{wmax}, 그리고 z_{wrand}를 갖는다. 그림 4에서 표현된 확률분포들과 지수확률분포를 조합하여 측정 모델에 적용할 확률분포를 만들어낸다. 측정 모델에 적용될 확률모델을 구하기 위해 사용되는 확률분포들의 적용도는 z_{wlong} 또는 z_{wshort}와 z_{whit} z_{wmax}, 그리고 z_{wrand}이다. 그림 4에 나타낸 확률분포들의 적용도를 모두 더한 값과 지수확률분포의 적용도인 z_{wlong}이나 z_{wshort}를 더하면 1이 된다. z_{wlong}은 송신기와 수신기로 분리되어 송신신호를 수신하는 센서에 적용하기 위한 적용도이다. z_{wshort}는 센서에서 송신한 신호가 장애물 등에 반사되어 돌아오는 신호를 수신하는 송수신 센서의 경우에 사용되는 적용도이다[15].

$$Z = [z_{whit}\ z_{wmax}\ z_{wrand}] \tag{3.2}$$

표 8. 공통 측정 모델 함수
Table 8. Common measurement model function

Algorithm MM_Common(z_t^k, z_t^{k*}, Z, σ_{hit}, z_{max})

1: *if* $0 \le z_t^k \le z_{\max}$

2: $p_{hit} = \frac{1}{\sqrt{2\pi\sigma_{hit}^2}} e^{-\frac{1}{2}\frac{(z_t^k - z_t^{k*})^2}{\sigma_{hit}^2}}$

3: *otherwise*
4: $p_{hit} = 0$
5: *if* $z \geq z_{\max}$
6: $p_{\max} = 1$
7: *otherwise*
8: $p_{\max} = 0$
9: *if* $0 \leq z_t^k \leq z_{\max}$
10: $p_{rand} = \frac{1}{z_{\max}}$
11: *otherwise*
12: $p_{rand} = 0$
13: *return* $(p_{hit} z_{whit} + p_{\max} z_{wmax} + p_{rand} z_{wrand})$

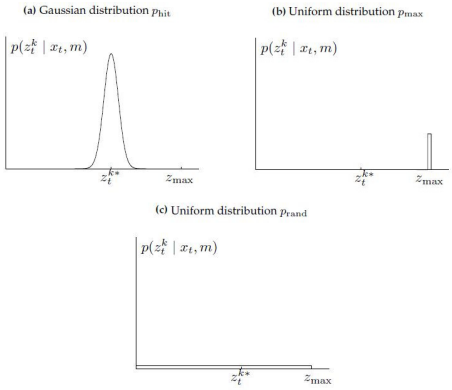

그림 4. 공통 측정 모델 함수에 사용 된 확률모델
Fig. 4 Distributions used common measurement model function

4.1 센서의 송수신 정보를 이용한 측정 모델

본 절에서 송수신 정보는 센서에서 송신된 후 장애물이나 목표물 등에 반사되어 센서에 수신되는 정보를 의미한다. 그림 5는 송수신 정보를 이용한 거리 측정 센서의 측정 모델에 이용되는 지수 확률분포를 나타낸다. 확률변수는 1에서 100까지이고 평균은 50, 센서 측정 최댓값은 100, 그리고 λ_{short}는 0.03을 입력하여 나타냈다. 표 9는 송수신 정보를 이용한 측정 모델 함수를 의사코드로 나타낸다. 함수의 명칭은 *MM_TR*라 정의한다. 함수의 입력정보는 센서의 측정 정보 z_t^k, 계산된 정보 z_t^{k*}, 4개 확률모델의 적용도인 Z와 z_{wshort}, 가우시안 확률분포에 사용될 표준편차 σ_{hit}, 지수 확률분포에 사용되는 λ_{short}, 그리고 센서의 측정 가능 최댓값 z_{max}를 입력받는다. 4개의 확률분포가 조합된 확률분포의 확률 값을 구하기 위해 *MM_TR* 함수 내에서 *MM_Common*함수를 호출한다. 함수의 출력은 조합된 확률분포의 확률 값을 반환한다.

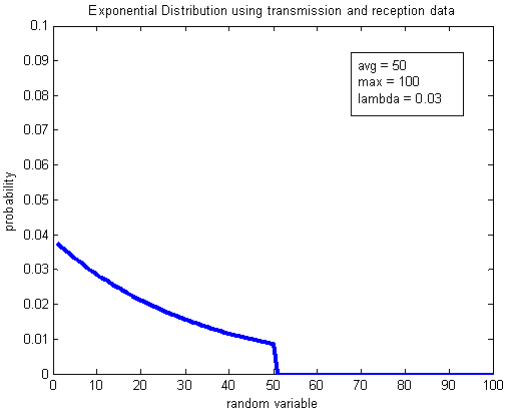

그림 5. 송수신 정보를 이용한 지수 확률분포
Fig. 5 Exponential distribution using transmission and reception data

표 9. 송수신 정보를 이용한 측정 모델 함수
Table 9. Measurement model function using transmission and reception data

Algorithm $MM\_TR(z_t^k, z_t^{k*}, Z, z_{wshort}, \sigma_{hit}, \lambda_{short}, z_{max})$
1: *if* $0 \leq z_t^k \leq z_t^{k*}$
2: $p_{short} = \frac{1}{1 - e^{-\lambda_{short} z_t^{k*}}} \lambda_{short} e^{-\lambda_{short} z_t^k}$
3: *otherwise*
4: $p_{short} = 0$
5: p = *call* $MM\_Common(z_t^k, z_t^{k*}, Z, \sigma_{hit}, z_{max})$
6: *return* $(p_{short} z_{w\_short}) + p$

4.2 센서의 수신된 정보를 이용한 측정 모델

본 절에서 수신된 정보는 특정 송신기로부터 센서에 수신된 고유의 신호나 정보를 의미한다. 대표적인 예로 초음파 위치측정 시스템이 있다[16]. 그림 6은 수신 정보를 사용 하는 측정 모델의 지수 확률분포를 나타낸다. 확률변수는 1에서 100까지이고 평균은 50, 센서 측정 최댓값은 100, 그리고 λ_{long}은 0.03을 입력하여 나타냈다. 표 10은 수신된 정보를 이용한 측정

모델 함수의 의사코드를 나타낸다. 함수의 명칭은 *MM_R*이라 정의한다. 함수의 입력정보는 센서의 측정 정보 z_t^k, 계산된 정보 z_t^{k*}, 4개 확률모델의 적용도인 Z와 z_{wlong}, 가우시안 확률분포에 사용될 표준편차 σ_{hit}, 지수 확률분포에 사용될 λ_{long}, 그리고 센서의 측정 가능 최댓값 z_{max}를 입력받는다. 함수의 출력은 조합된 확률분포의 확률 값을 반환한다.

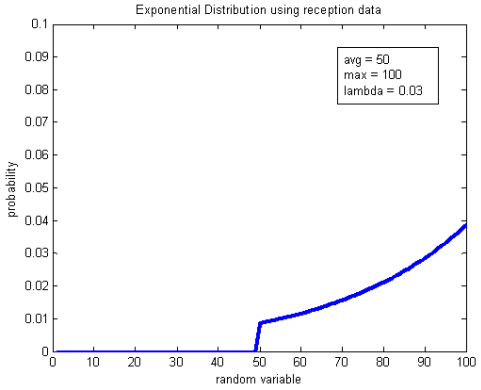

그림 6. 수신 정보를 이용한 지수 확률분포
Fig. 6 Exponential distribution using reception data

표 10. 수신 정보를 이용한 측정 모델 함수
Table 10. Measurement model function using reception data

Algorithm $MM\_R(z_t^k, z_t^{k*}, Z, z_{w\_long}, \sigma_{hit}, \lambda_{long}, z_{max})$

1: *if* $z_t^{k*} \le z_t^k \le z_{\max}$
2: $p_{short} = \dfrac{1}{1-e^{-\lambda_{long} z_t^{k*}}} \lambda_{long} e^{-\lambda_{long}(z_{\max} - z_t^k)}$
3: *otherwise*
4: $p_{hit} = 0$
5: p = *call* $MM\_Common(z_t^k, z_t^{k*}, Z, \sigma_{hit}, z_{max})$
6: *return* $(p_{short} z_{w\_long}) + p$

Ⅴ. 방향 관련 연산

이동 로봇에서는 로봇의 방향을 $-\pi$에서부터 π까지 표현가능하다. 기준이 되는 축을 중심으로 반시계 방향의 π까지는 양수로, 시계방향의 π까지는 음수로 표현한다. 그래서 로봇에 사용되는 방향 값은 연산이 편리하지 않다. 로봇 방향의 연산을 위해 본장에서는 두 방향의 합과 차를 구하는 함수와 방향들의 대푯값을 구하는 함수를 설명한다.

5.1 두 방향의 합과 차

표 11은 두 방향의 합을 구하는 의사코드를 나타낸다. 두 방향의 합은 두 방향을 합하고 합한 값을 2π로 나눈다. 나눈 나머지 값이 π보다 큰 값이면 -2π를 더해 0에서 $-\pi$까지 범위의 값을 갖게 한다. $-\pi$보다 작은 값이라면 2π를 더하여 0에서 π까지 범위를 갖게 한다. 두 방향의 합을 구하는 함수의 명칭은 *AngleSum*라 정의한다. 입력정보는 두 방향을 입력받고, 함수의 출력은 합한 결과를 반환한다.

표 11. 방향의 합을 구하는 함수
Table 11. Sum function of direction

Algorithm $AngleSum(\theta_1, \theta_2)$

1: $result = mod(\theta_1 + \theta_2, 2\pi)$
2: *if* $result > \pi$
3: $result = -2\pi + result$
4: *else if* $result < -\pi$
5: $result = 2\pi + result$
6: *return result*

표 12는 방향의 차를 구하는 함수의 의사코드를 나타낸다. 함수의 명칭은 *AngleDifference*라 정의한다. 함수의 출력은 입력된 두 방향의 사잇각을 반환한다. 반환되는 사잇각은 입력받은 두 방향의 사잇각 중 작은 사잇각을 의미한다. 함수의 출력 값은 첫 번째 입력정보를 기준으로 반환 값의 부호가 달라진다[5].

표 12. 방향의 차를 구하는 함수
Table 12. Difference function of direction

Algorithm $AngleDifference(\theta_1, \theta_2)$

1: $result = mod(\theta_1 - \theta_2 + \pi, 2\pi)$
2: $result = mod(result - \pi, 2\pi)$
3: *if* $result > \pi$ *and* $result < 2\pi$
4: $result = -2\pi + result$
5: *else if* $result > -2\pi$ *and* $result < -\pi$
6: $result = 2\pi + result$
7: *return result*

5.2 방향들의 대푯값

표 13은 방향들의 대푯값을 구하는 의사코드를 나타낸다. 함수의 명칭은 *AngleAvg*라 정의한다. 함수의 입력정보는 방향들을 입력받는 θ_n과 방향의 개수인 N을 입력받는다. 입력받은 방향들의 대푯값을 구하기 위해 크기가 1인 단위벡터들의 합을 구하는 방법을 이용한다. 본 연구에서는 *AngleAvg* 함수 내에 임의의 두 변수를 만들었다. 방향들의 sin값을 합한 결과를 갖는 변수와 다른 하나는 방향들의 cos값을 합한 결과를 갖는 변수이다. sin의 모든 합, cos의 모든 합을 arctan를 이용하여 평균 방향을 구한다. 피타고라스 정리를 이용하여 방향들의 결과 값에 대한 벡터의 크기를 구할 수 있다. 획득한 벡터의 크기에 입력받은 방향의 총 개수로 나눈 값은 입력된 방향들이 평균 방향에 밀집된 분포도와 같다. 모든 방향이 같은 방향이라면 분포도는 1이 되고, 모든 방향이 고르게 분포되어 있다면 0에 가까운 값을 얻는다. 함수의 출력으로는 대표 방향과 분포도를 반환한다.

표 13. 방향의 대푯값을 구하는 함수
Table 13. Representative value function of multidirection

Algorithm AngleAvg(θ_n, N)

1: *for all* θ_i *do*
2: $res\_s = res\_s + \sin\theta_i$
3: $res\_c = res\_c + \cos\theta_i$
4: *end for*
5: $AvgAngle = \arctan\left(\frac{res\_s}{res\_c}\right)$
6: $distribution = \frac{\sqrt{res\_s^2 + res\_c^2}}{N}$
7: *return AvgAngle, distribution*

Ⅵ. 실 험

본 실험에서는 개발된 함수를 적용한 모의실험과 실제 로봇에 함수를 적용한 실험결과를 보인다. 표 14부터 표 16에 나타낸 값들을 *DR_DD* 함수, *DR_BF* 함수, 그리고 *DR_BR* 함수의 입력정보에 사용했다. 2번째부터 100번째까지 함수 실행 시 이전 위치 정보는 이전 실행으로 획득한 출력 값의 추정 위치 정보를 사용했다. 입력정보의 속도정보 또는 앞바퀴와 로봇몸체의 사잇각은 표에 기록된 값을 적용했다. 그림 7은 *DR_DD*함수, 그림 8은 *DR_BF*함수, 그리고 그림 9는 *DR_BR*함수를 이용한 모의실험으로 획득한 추정 위치 정보를 나타낸다.

표 14. DR_DD함수의 입력정보
Table 14. Input parameters of DR_DD function

x	0 m
y	0 m
θ	0 rad
v	1 m/s
w	0.175 rad/s
Δt	0.1 s

표 15. DR_BF함수의 입력정보
Table 15. Input parameters of DR_BF function

x	0 m
y	0 m
θ	0 rad
v	1 m/s
ϕ	0.175 rad
ℓ	1 m
Δt	0.1 s

표 16. DR_BR함수의 입력정보
Table 16. Input parameters of DR_BR function

x	0 m
y	0 m
θ	0 rad
v	1 m/s
ϕ	0.175 rad
ℓ	1 m
Δt	0.1 s

그림 6부터 그림 8에 나타낸 적색 선은 추정된 로봇의 경로를 표현하고 청색 원은 추정된 로봇을 나타낸다. 입력된 매개변수의 큰 차이가 없어 그림 6과 그림 7, 그림 8 모두 유사해 보인다. 실제로 연산하여 비교해 보면 세 함수의 결과 값의 큰 차이가 없다. 삼각함수 연산 결과에 의한 차이가 매우 적기 때문이

다. 실제 연산결과와 함수 실행결과가 큰 차이를 보이지 않으므로 추측 항법을 사용한 모의실험이나 실제 로봇에 구현된 함수를 적용할 수 있음을 보였다.

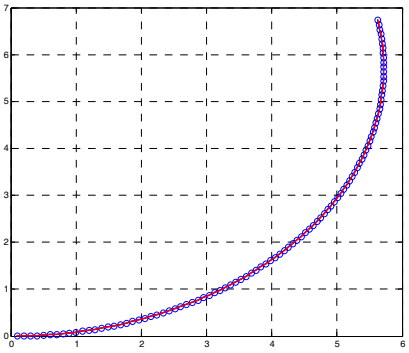

그림 7. DR_DD함수 실행 결과
Fig. 7 Result of DR_DD function

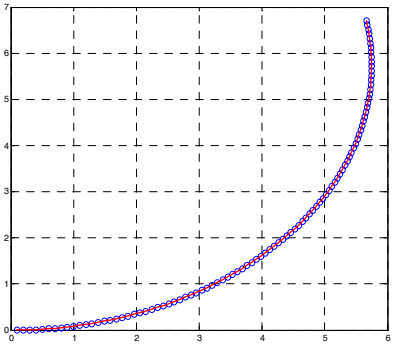

그림 8. DR_BF함수 실행 결과
Fig. 8 Result of DR_BF function

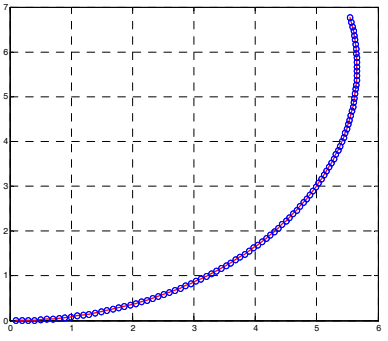

그림 9. DR_BR함수 실행 결과
Fig. 9 Result of DR_BR function

실제 실험은 구현된 함수들을 사용해 몬테카를로(*Monte Carlo*)알고리즘을 로봇에 적용했다[17-20]. 사용된 함수는 *DR_DD, NormalDistribution, MM_DD, MM_RF, AngleSum, AngleDifference, AngleAve*이다. 그림 10의 왼쪽 사진을 보면 분홍색 원은 추정된 로봇의 위치이고 파란색 원들은 파티클(*particle*)들을 나타낸다. 실제 로봇에 함수들을 적용해 위치 추정 알고리즘을 구현하고 로봇의 위치가 추정됨을 확인 가능하다. 이로써 개발한 함수들을 실제 로봇에 적용하여 위치 추정을 구현할 수 있음을 보였다.

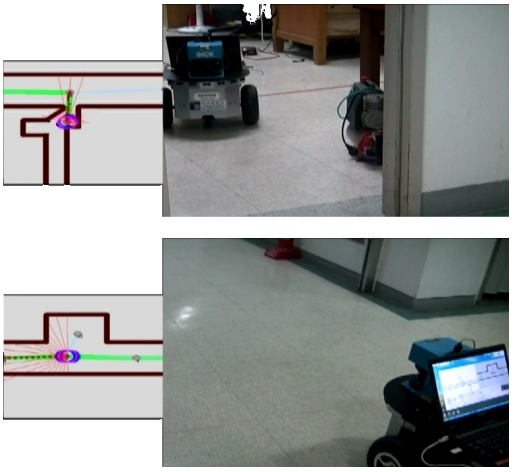

그림 10. 이동 로봇에 함수 적용 및 실험
Fig. 10 Applying function to mobile robot and experiment

본 실험에서의 모의실험 결과들은 이론에 부합하는 결과 데이터들을 보여주었다. 모의실험을 통해 본 연구로 개발된 함수들의 이용에 문제점이 없음을 보였다. 또한 실제 로봇의 몬테카를로 위치 추정 알고리즘에 적용한 실험을 통해 개발된 함수 라이브러리들을 검증했다.

Ⅶ. 결 론

본 연구에서는 이동로봇 자율주행 요소 기술 중 위치 추정 기술을 구현하기 위한 함수들을 개발했다. 본 연구에서는 이동 로봇의 위치 추정을 위한 추측 항법, 운동 모델, 그리고 외부 환경 정보를 인지하기 위한

측정 모델용 함수를 구현했다. 추측 항법과 운동 모델 함수는 차륜 구동 로봇과 이륜차 로봇을 모델로 했다.

본 실험에서는 함수들을 모의실험과 실제 로봇의 몬테카를로 알고리즘에 적용하여 위치 추정에 사용가능함을 보였다. 이로써 위치 추정 알고리즘에 필요한 기능들을 구현된 함수를 사용하여 접목시킬 수 있음을 보였다.

기존에 사용 되고 있는 라이브러리와는 달리 각 함수의 설명을 자세히 서술하여 사용자가 쉽게 이용가능하고, 실제 로봇에 적용하여 알고리즘의 구현이 가능하다. 또한 구현한 함수의 배경이론을 잘 설명하고 있어 사용자가 원하는 함수를 찾아 사용하기에 편리하다. 본 논문을 읽고 숙지한 후 함수를 사용한다면 이동 로봇의 위치 추정 알고리즘 구현 시 시간소모를 줄일 수 있을 것으로 사료된다.

감사의 글

본 논문은 교육과학기술부, 한국연구재단의 2010년 지역혁신인력양성사업(과제명 : 로봇의 자율주행 요소 기술 상용화 및 인력양성, 과제관리번호 : 2010-04-대-01-016)의 지원에 의해 이루어짐.

참고 문헌

[1] 진조철, “위치 인식 시스템 개발 동향 소개”, 한국통신학회지, 25권, 4호, pp. 5-10, 2008.
[2] 유원필, 최성록, 이재영, 박승환, “로봇주행 기술 및 표준화 동향”, 전자통신동향분석, 26권, 6호, pp. 108-119, 2011.
[3] Jose Luls Blanco Claraco, “Development of Scientific Applications with the Mobile Robot Programming Toolkit”, The MRPT reference book, 2010.
[4] http://www.mrpt.org/
[5] Peter Corke, Robotics, “Vision and Control Fundamental Algorithms in MATLAB”, Springer, 2011.
[6] http://www.petercorke.com/RVC/
[7] 유원필, “설명회자료-범용 로봇 주행 라이브러리 uRON”, 한국전자통신연구원, 2009.
[8] http://carmen.sourceforge.net/home.html
[9] http://www.kartorobotics.com/
[10] 정석기, “이동 로봇의 위치 추정을 위한 함수 라이브러리 개발”, 조선대학교 석사 학위 논문, 2013.
[11] Julius Maximilian Univeresitat Wurzburg, “Kinematics of a car-like moile robot”, 2003.
[12] 정문수, 안성수, “Autonomous Naviagation system for Power Wheelchair System”, 한국전자통신학회논문지, 4권, 1호, pp. 37-45, 2009.
[13] Bruno Siciliano, Lorenzo Sciavicco, Luigi Villani, Giuseppe Oriolo, “Robotics Modelling”, Planning and Control, Springer, 2009.
[14] 남택근, 김철승, “비 홀로노믹 구속조건을 이용한 수중 이동체의 자세제어 연구,” 한국항해항만학회, 28권, 6호, pp. 469-474, 2004.
[15] Sebastian Thrun, Wolfram Burgard, Dieter Fox, “Probabilistic Robotics”, The MIT Press, 2005.
[16] 윤강섭, “초음파 위성 시스템을 위한 개선된 위치추정 알고리즘”, 한국전자통신학회논문지, 6권, 5호, pp. 775-781, 2011.
[17] 노성우, 고낙용, 김태균, “위치 추정, 충돌 회피, 동작 계획이 융합된 이동로봇의 자율주행 기술 구현”, 한국전자통신학회논문지, 6권, 1호, pp. 148-156, 2011.
[18] 노성우, 김태균, 고낙용, 배영철, “이동로봇의 GPS위치 정보 보정을 위한 파티클 필터”, 한국전자통신학회논문지, 7권, 2호, pp. 381-389, 2012.
[19] 김태균, 고낙용, 노성우, “초음파 비이컨을 사용한 이동로봇 실내 주행용 파티클 필터 SLAM”, 한국전자통신학회논문지, 7권, 2호, pp. 391-399, 2012.
[20] Hester, P., Stone, P., “Navigation information and line observation for Monte Carlo Localization”, Proc. 2008 IEE Int. Conf. Robotics and Automation, pp. 2764-2769, 2008.

저자 소개

정석기(Seok-Ki Jeong)

2011년 호남대학교 전자공학과 졸업 (공학사)
2011년~현재 조선대학교 대학원 제어계측공학과 공학석사 과정
※ 관심분야 : 이동 로봇, 자율주행

고낙용(Nak-Yong Ko)

1985년 서울대학교 제어계측공학과 졸업(공학사)
1987년 서울대학교 대학원 제어계측공학과 졸업(공학석사)
1993년 서울대학교 대학원 제어계측공학과 졸업(공학박사)
1997～1998, 2004～2005 미국 Carnegie Mellon Univ. Visiting research scientist
1992년～현재 조선대학교 제어계측로봇공학과 교수
※ 관심분야 : 지상로봇과 수중로봇의 자율주행

김태균(Tae-Gyun Kim)

2007년 조선대학교 제어계측공학과 졸업(공학사)
2009년 조선대학교 대학원 제어계측공학과 졸업(공학석사)
2009년～현재 조선대학교 대학원 제어계측학과 공학박사 과정
※ 관심분야 : 이동 로봇, 수중로봇, 자율주행